CSE 559A: Computer Vision

[credit: danjodon.deviantart.com]

Fall 2017: T-R: 11:30-1pm @ Lopata 101

Instructor: Ayan Chakrabarti (ayan@wustl.edu).
Staff: Abby Stylianou (abby@wustl.edu), Jarett Gross (jarett@wustl.edu)

http://www.cse.wustl.edu/~ayan/courses/cse559a/

Oct 24, 2017

GENERAL

- Problem Set 3 Due Thursday.
- Project Proposals Due Sunday.
- Regular Office Hours this week.

GLOBAL OPTIMIZATION (RECAP)

$$d = \arg\min_d \sum_n C[n, d[n]] + \lambda \sum_{(n,n')\in\mathbf{E}} S(d[n], d[n'])$$

- Discrete optimization of **disparity map** d, each $d[n] \in \{0, 1, \ldots D-1\}$
- $C[n, d[n]]$ comes from our matching cost. How well $L[x, y]$ matches $R[x - d[x, y], y]$.
- E is the set of all pairs of "neighboring" pixel locations.

- S is a function that indicates a preference for $d[n]$ and $d[n']$ to be the same.

- Example 1: 0 if $d[n'] = d[n]$, 1 otherwise.
- Example 2: $|d[n'] - d[n]|$
- Example 3:
  - 0 if $d[n'] = d[n]$
  - T_1 if $|d[n'] - d[n]| < \epsilon$
  - T_2 otherwise.

GLOBAL OPTIMIZATION

$$d = \arg\min_d \sum_n C[n, d[n]] + \lambda \sum_{(n,n')\in\mathbf{E}} S(d[n], d[n'])$$

Iterated Conditional Modes

- Begin with $d_0 = \arg\min_d C[n, d[n]]$
- At each iteration t, compute d_{t+1} from d_t, by solving
  for each pixel in d_{t+1} assuming neighbors have values from d_t.

$$d_{t+1}[n] = \arg\min_{d_n} C[n, d_n] + \lambda \sum_{(n,n')\in\mathbf{E_n}} S(d_n, d_t[n'])$$

Does it converge ?

- No Guarantee.
  A modified version would converge to a local minima if in each iteration, we only updated one pixel.

GLOBAL OPTIMIZATION

$$d = \arg\min_d \sum_n C[n, d[n]] + \lambda \sum_{(n,n') \in \mathbf{E}} S(d[n], d[n'])$$

Iterated Conditional Modes (slow!)

- Begin with $d_0 = \arg\min_d C[n, d[n]]$
- At each iteration t, compute d_{t+1} from d_t, by solving for **one** pixel in d_{t+1} assuming neighbors have values from d_t.

$$d_{t+1}[n_{t+1}] = \arg\min_{d_n} C[n_{t+1}, d_n] + \lambda \sum_{(n_{t+1}, n') \in \mathbf{E_{n_{t+1}}}} S(d_n, d_t[n'])$$

Does it converge ?

- No Guarantee.
  A modified version would converge to a local minima if in each iteration, we only updated one pixel n_t at iteration t.

GLOBAL OPTIMIZATION

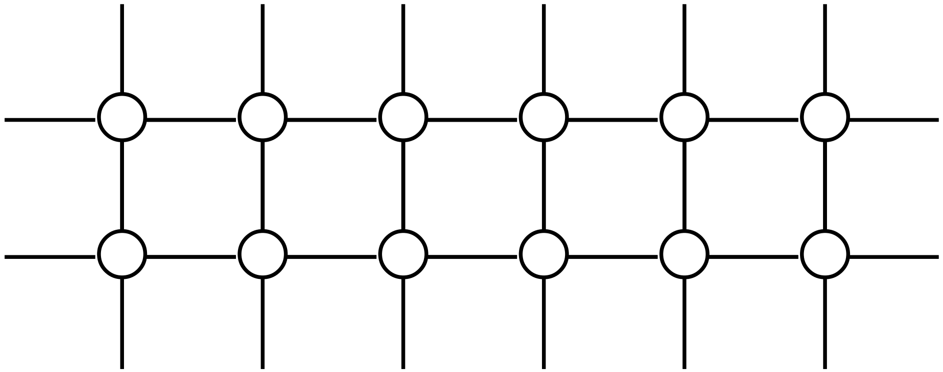
$$d = \arg\min_d \sum_n C[n, d[n]] + \lambda \sum_{(n,n')\in\mathbf{E}} S(d[n], d[n'])$$

- These kind of cost functions / optimization problems are quite common in vision.
- The cost can be interpreted as a log probability distribution:

$$p(d) \propto \prod_n \exp(-C[n, d[n]]) \prod_{(n,n')\in\mathbf{E}} \exp\left(-\lambda S(d[n], d[n'])\right)$$

- Joint distribution over all the $d[n]$ values.

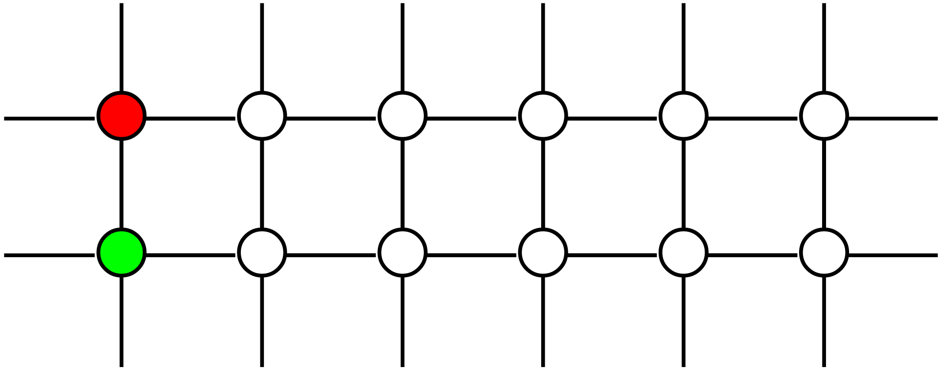
GLOBAL OPTIMIZATION



$$p(d) \propto \prod_{n} \exp(-C[n, d[n]]) \prod_{(n,n')\in\mathbf{E}} \exp\left(-\lambda S(d[n], d[n'])\right)$$

- Joint distribution over all the $d[n]$ values.

GLOBAL OPTIMIZATION



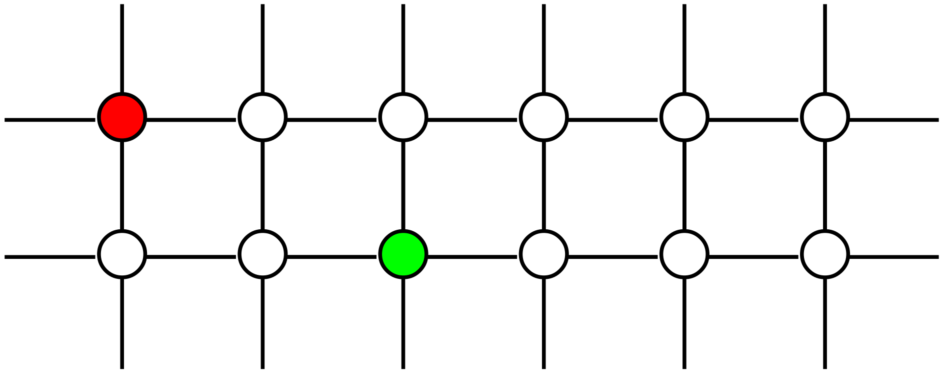
$$p(d) \propto \prod_n \exp(-C[n, d[n]]) \prod_{(n,n')\in\mathbf{E}} \exp\big(-\lambda S(d[n], d[n'])\big)$$

Question: Are $d[n]$ and $d[n']$ independent if:

- If $(n, n') \in \mathbf{E}$ -- pixels are neighbors ?

Reminder: Two variables are independent if we can express their joint distribution as a product of distributions on each variable.

GLOBAL OPTIMIZATION

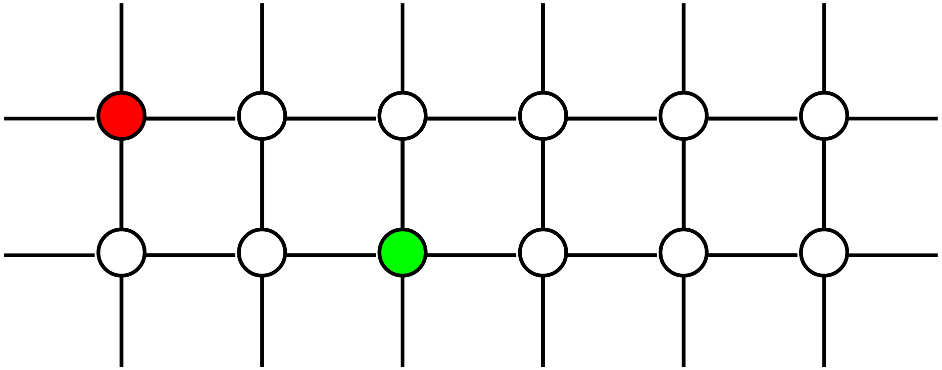


$$p(d) \propto \prod_n \exp(-C[n, d[n]]) \prod_{(n,n') \in \mathbf{E}} \exp\big(-\lambda S(d[n], d[n'])\big)$$

Question: Are $d[n]$ and $d[n']$ independent if:

- If $(n, n') \in \mathbf{E}$ -- pixels are neighbors. No
- If $(n, n') \notin \mathbf{E}$ -- pixels are not neighbors ?

GLOBAL OPTIMIZATION

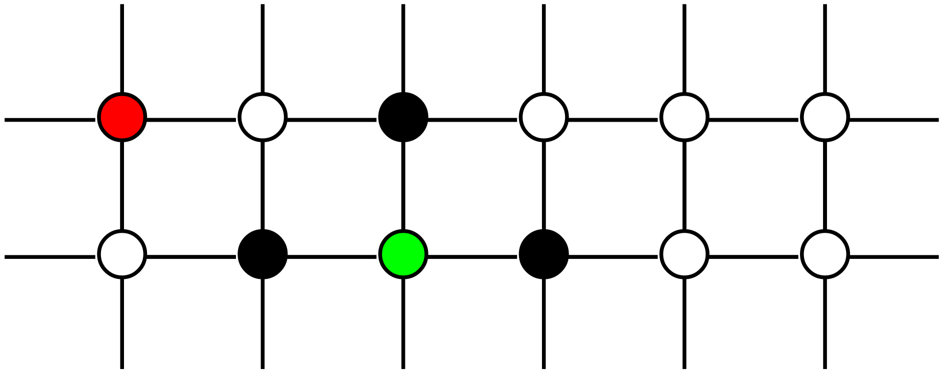


$$p(d) \propto \prod_{n} \exp(-C[n, d[n]]) \prod_{(n,n')\in\mathbf{E}} \exp\left(-\lambda S(d[n], d[n'])\right)$$

Question: Are $d[n]$ and $d[n']$ independent if:

- If $(n, n') \notin \mathbf{E}$ -- pixels are not neighbors ? NO. Unless n, n' are parts of disconnected components of graph.

GLOBAL OPTIMIZATION



$$p(d) \propto \prod_{n} \exp(-C[n, d[n]]) \prod_{(n,n')\in\mathbf{E}} \exp\big(-\lambda S(d[n], d[n'])\big)$$

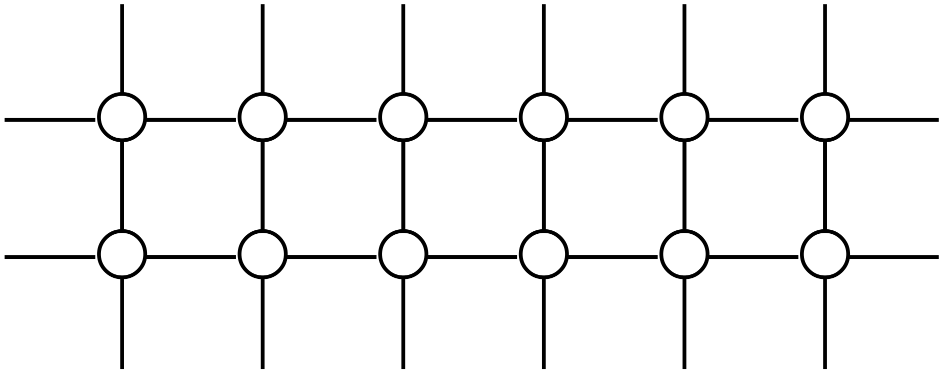
Question: Are $d[n]$ and $d[n']$ independent if:

- If $(n, n') \notin \mathbf{E}$, "conditioned" on all the neighbors of n being observed. $p(d[n], d[n'] | \{d[n'']\})$

YES. This is the Markov property. And these kinds of graphical models are called Markov random fields.

Graph structure encodes "conditional independence".

GLOBAL OPTIMIZATION



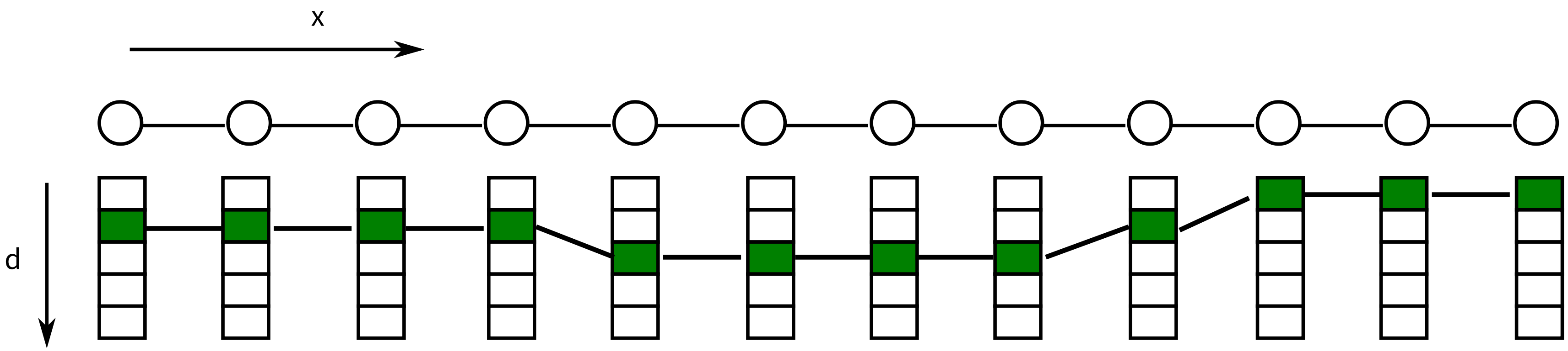
$$d = \arg\max_d p(d) = \arg\min_d \sum_n C[n, d[n]] + \lambda \sum_{(n,n')\in\mathbf{E}} S(d[n], d[n'])$$

GLOBAL OPTIMIZATION

$$d = \arg\min_{d} \sum_{n} C[n, d[n]] + \lambda \sum_{(n,n') \in \mathbf{E}} S(d[n], d[n'])$$

- Iterated Conditional Modes really slow.
- No guaranteed solution for arbitrary graphs.
- But could solve it we our graph were a chain (or more generally a tree).

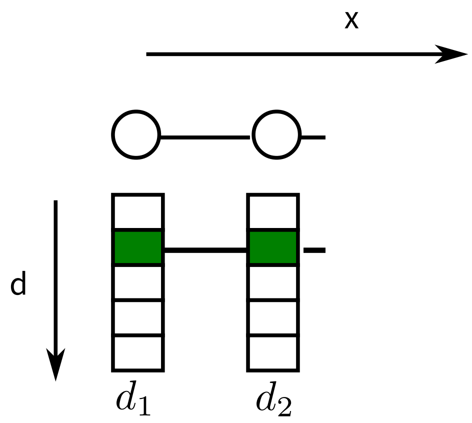
GLOBAL OPTIMIZATION



$$\sum_x C[x, d[x]] + \lambda \sum_x S(d[x], d[x+1])$$

The total cost of those blocks and the edges was the least.

GLOBAL OPTIMIZATION



Say we only had two nodes:

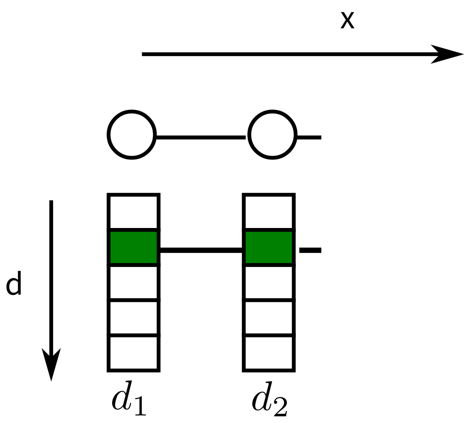
$$d_1, d_2 = \arg\min_{d_1\, d_2} C[1, d_1] + C[2, d_2] + \lambda S(d_1, d_2)$$

$$d_2 = \arg\min_{d_2} \min_{d_1} C[1, d_1] + C[2, d_2] + \lambda S(d_1, d_2)$$

This is the d_2 corresponding to the optimal path

$$\sum_x C[x, d[x]] + \lambda \sum_x S(d[x], d[x+1])$$

GLOBAL OPTIMIZATION



Say we only had two nodes:

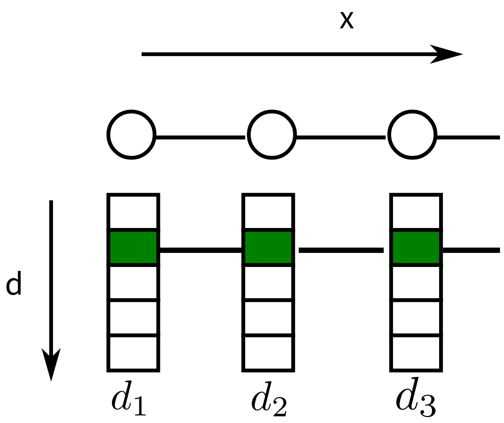
$$d_1, d_2 = \arg\min_{d_1\, d_2} C[1, d_1] + C[2, d_2] + \lambda S(d_1, d_2)$$

$$d_2 = \arg\min_{d_2} C[2, d_2] + \boxed{\min_{d_1} \left(C[1, d_1] + \lambda S(d_1, d_2)\right)}$$

This is a function of d_2
or a table of values for each possible value of d_2

$$\sum_x C[x, d[x]] + \lambda \sum_x S(d[x], d[x+1])$$

GLOBAL OPTIMIZATION



$$d_1, d_2, d_3 = \arg\min C[1, d_1] + C[2, d_2] + C[3, d_3] + \lambda S(d_1, d_2) + \lambda S(d_2, d_3)$$

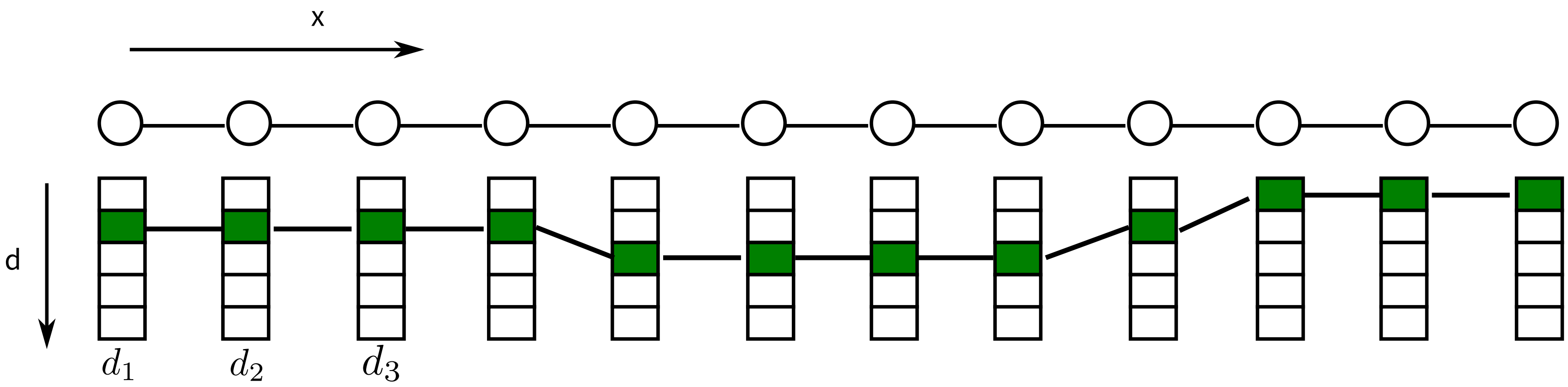
$$d_3 = \arg\min_{d_3} C[3, d_2] + \min_{d_2} \left[\lambda S(d_2, d_3) + C[2, d_2] + \underline{\min_{d_1} [\lambda S(d_1, d_2) + C[1, d_1]]} \right]$$

This is precisely what we computed for the 2 node case

Also note that once you have this, you don't care about what the value of d_1 was in the inner minimization.

$$\sum_x C[x, d[x]] + \lambda \sum_x S(d[x], d[x+1])$$

GLOBAL OPTIMIZATION

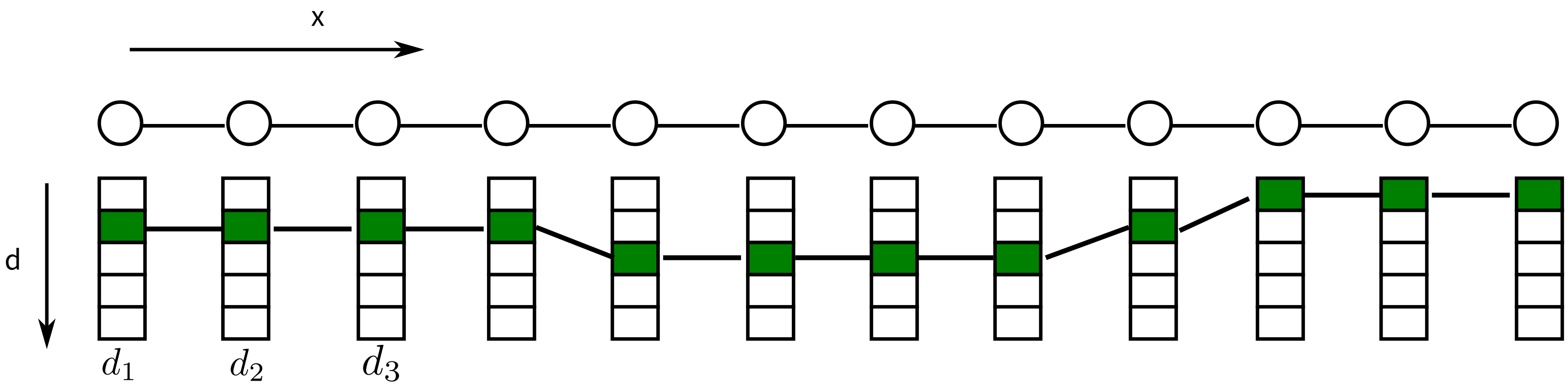


$$d_1, d_2, d_3 = \arg\min C[1, d_1] + C[2, d_2] + C[3, d_3] + \lambda S(d_1, d_2) + \lambda S(d_2, d_3)$$

$$d_3 = \arg\min_{d_3} \underbrace{C[3, d_2] + \min_{d_2} \left[\lambda S(d_2, d_3) + \underbrace{C[2, d_2] + \min_{d_1} [\lambda S(d_1, d_2) + C[1, d_1]]}_{\bar{C}[2, \cdot]} \right]}_{\bar{C}[3, \cdot]}$$

$$\bar{C}[x+1, d] = C[x+1, d] + \min_{d'} \lambda S(d, d') + \bar{C}[x, d']$$

GLOBAL OPTIMIZATION



We go from left to right, and doing an arg min on the last $\bar{C}$ gives us the disparity of the last node.

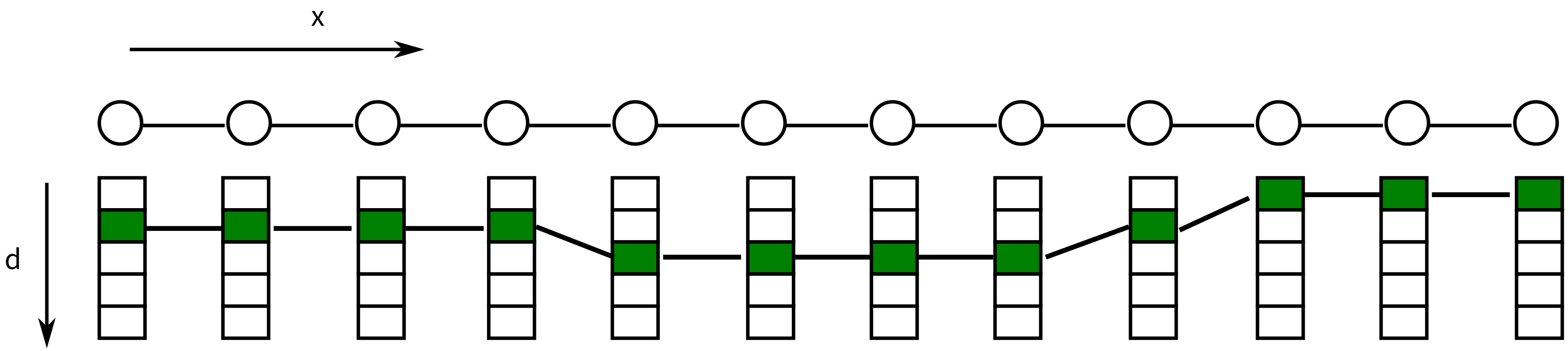
And then we backtrack to find the full chain.

Store best d' for each d.

$$z[x+1,d] = \arg\min_{d'} \lambda S(d,d') + \bar{C}[x,d']$$

$$\bar{C}[x+1,d] = C[x+1,d] + \min_{d'} \lambda S(d,d') + \bar{C}[x,d']$$

GLOBAL OPTIMIZATION



Forward

$$\bar{C}[0,d] = C[0,d]$$

$$z[x+1,d] = \arg\min_{d'} \lambda S(d,d') + \bar{C}[x,d']$$

$$\bar{C}[x+1,d] = C[x+1,d] + \min_{d'} \lambda S(d,d') + \bar{C}[x,d']$$

Backward

$$d[x_{end}] = \arg\min_{d} \bar{C}[x_{end},d]$$

$$d[x] = z[x+1, d[x+1]]$$

GLOBAL OPTIMIZATION

We could apply this on individual epipolar lines.

GLOBAL OPTIMIZATION

- That's why we want to use a full 2D grid.
- But forward-backward only works on chains (or graphs without cycles).

One flavor of approximate algorithms apply the same idea of forming a $\bar{C}[x, d]$

- TRW-S
- Loopy Belief Propagation
- SGM

GLOBAL OPTIMIZATION

Semi-Global Matching

$$\bar{C}[x, d] = C[x, d] + \min_{d'} \bar{C}[x-1, d'] + \lambda S(d, d')$$

This is going left to right in the horizontal direction.

Idea: Compute different $\bar{C}$ along different directions ...

and average.

GLOBAL OPTIMIZATION

Semi-Global Matching

$$\begin{aligned}
\bar{C}_{lr}[n,d] &= C[n,d] + \min_{d'} \bar{C}_{lr}[n - [1,0]^T, d'] + \lambda S(d,d') \\
\bar{C}_{rl}[n,d] &= C[n,d] + \min_{d'} \bar{C}_{rl}[n + [1,0]^T, d'] + \lambda S(d,d') \\
\bar{C}_{du}[n,d] &= C[n,d] + \min_{d'} \bar{C}_{du}[n - [0,1]^T, d'] + \lambda S(d,d') \\
\bar{C}_{ud}[n,d] &= C[n,d] + \min_{d'} \bar{C}_{ud}[n + [0,1]^T, d'] + \lambda S(d,d')
\end{aligned}$$

$$d[n] = \arg\min_{d} \bar{C}_{lr}[n,d] + \bar{C}_{rl}[n,d] + \bar{C}_{ud}[n,d] + \bar{C}_{du}[n,d]$$

GLOBAL OPTIMIZATION

Semi-Global Matching